



US 20070300135A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2007/0300135 A1  
Kamiya (43) Pub. Date: Dec. 27, 2007(54) ERROR CORRECTION ENCODING  
APPARATUS AND ERROR CORRECTION  
ENCODING METHOD USED THEREIN

## Publication Classification

(51) Int. Cl.  
H03M 13/19 (2006.01)  
G06F 11/10 (2006.01)  
(52) U.S. Cl. 714/781; 714/711

(75) Inventor: Norifumi Kamiya, Tokyo (JP)

Correspondence Address:  
YOUNG & THOMPSON  
745 SOUTH 23RD STREET  
2ND FLOOR  
ARLINGTON, VA 22202 (US)

## (57) ABSTRACT

An error correction encoding apparatus wherein the apparatus structure is simple; an iterative decoding is used to achieve a decoding with a close-to-optimum precision; and a simple mathematical expression is used to perform an evaluation of the characteristic of an error floor area without using any computer experiments. In a polynomial multiplying block 1, (n-1)-th-order polynomial multiplying units (12-1 to 12-(m-1)) further divide an information bit string, which has been blocked for an error correction encoding, into (m-1) blocks having a length n and a single block having a length (n-r) (where m and n represent integers equal to or greater than two and where r represents an integer between 1 and n inclusive); receive blocks, which have the length n, of divided information bit strings; and output a series having the same length. An r-th-order polynomial dividing unit 2 receives an addition of the outputs from the respective (n-1)-th-order polynomial multiplying units (12-1 to 12-(m-1)) and also receives a block having the length (n-r), and outputs a redundant bit series having a length r.

(73) Assignee: NEC CORPORATION, TOKYO (JP)

(21) Appl. No.: 11/792,692

(22) PCT Filed: Nov. 29, 2005

(86) PCT No.: PCT/JP05/21909

§ 371(c)(1),  
(2), (4) Date: Jun. 8, 2007

(30) Foreign Application Priority Data

Dec. 15, 2004 (JP) 2004-362135

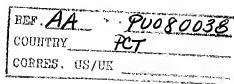
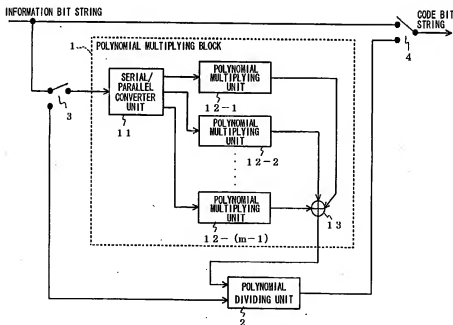


FIG. 1

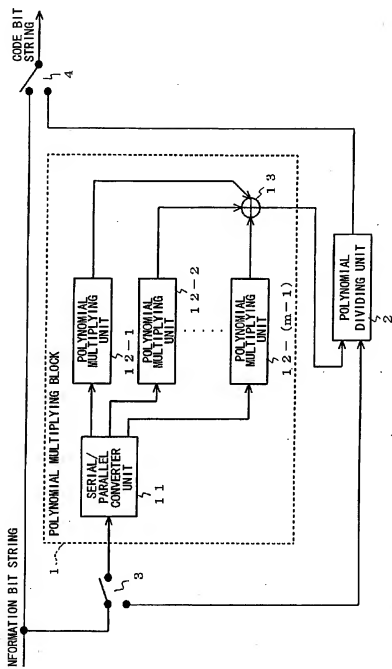
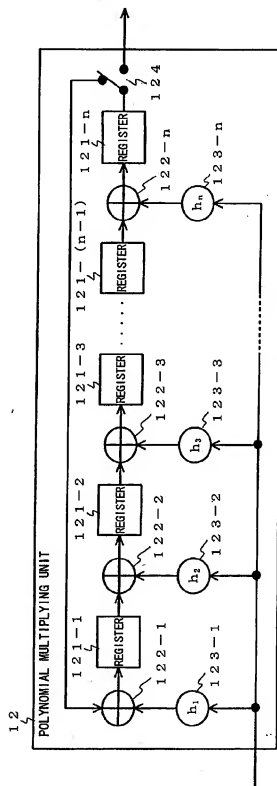


FIG. 2



**FIG. 3**

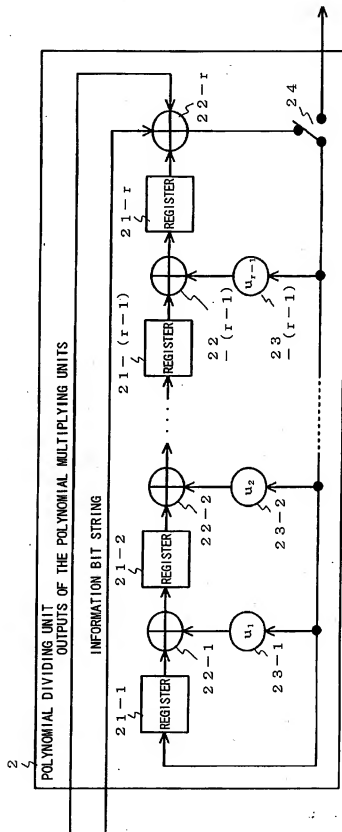


FIG. 4

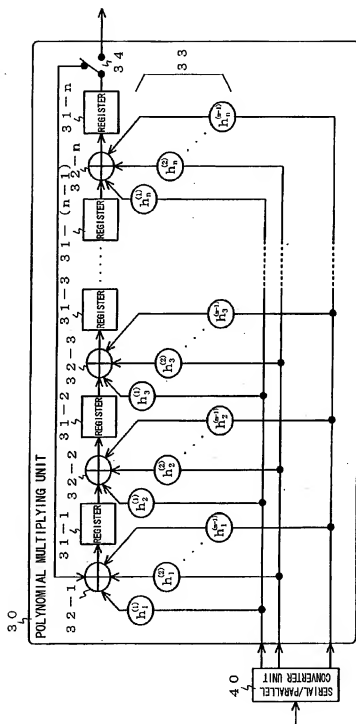


FIG. 5

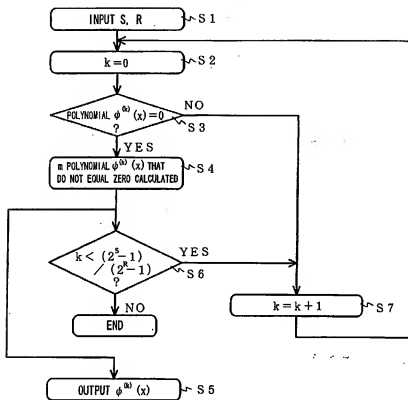


FIG. 6

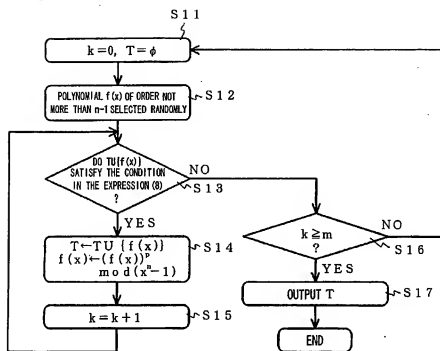
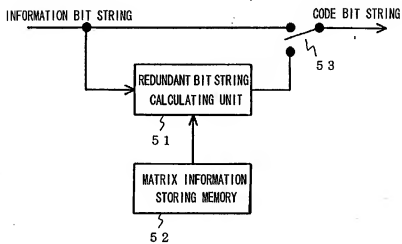


FIG. 7

RELATED ART





# ERROR CORRECTION ENCODING APPARATUS AND ERROR CORRECTION ENCODING METHOD USED THEREIN

## TECHNICAL FIELD

[0001] The present invention relates to an error correction encoding apparatus and an error correction encoding method used therein, and particularly to a block error correction encoding system wherein an information series is divided into blocks having a predetermined length and a redundant series is independently added to each block, a low-density parity-check (LDPC) encoding method used in a circuit therein, and an apparatus thereof.

## BACKGROUND ART

[0002] Error correction encoding technologies achieving large coding gain have been introduced to satellite communications and mobile telecommunication systems in order to meet the demands regarding the system structure such as reducing required power and the size of antennas. A low-density parity-check code is known as an error correction code achieving very large coding gain, and has been used in various communication systems and storage units such as magnetic storage units.

[0003] The low density parity check code does not refer to one particular error correction encoding method, but it is a general term for error correction codes characterized by a sparse check matrix (most elements in the matrix are 0, and the number of "1" elements is very few). By selecting a sparse check matrix and using an iterative decoding method such as the sum-product algorithm, the low-density parity-check code can realize an error correction encoding system capable of achieving very large coding gain close to the theoretical limit (for instance refer to Non-Patent Documents 1 and 2).

[0004] Technical problems regarding the low-density parity-check code are as follows. The amount of calculation required for the encoding method (the method for calculating a redundant bit sequence from an information bit sequence) is large, and it is difficult to evaluate the performance of the error-rate characteristics (the obtained coding gain) in an area where the error probability is low, particularly an error floor area. In a most typical encoding apparatus in which the encoding system is constituted by matrix multiplication by a generator matrix of the code, the number of exclusive-OR operations required is proportional to the square of the code length.

[0005] Further, when an encoding apparatus is constituted by a check matrix of the code, the check matrix is transformed by means of elementary matrix transformations so that it partially forms a diagonal matrix such as an expression (1) and the apparatus is realized by the transformed check matrix.

[Expression 1]

$$\begin{bmatrix} 1 & & 0 \\ A & I & \\ 0 & & I \end{bmatrix} \quad (1)$$

[0006] More concretely, when the part indicated by A in the expression (1) is an  $r \times k$  matrix (where  $r$  and  $k$  represent

positive integers), and  $c_1, c_2, \dots, c_k$  is an information bit series of  $k$  bits, each bit  $p_i$  (where  $i$  represents an integer between 1 and  $r$  inclusive) of a redundant bit string  $p_1, p_2, \dots, p_r$  of  $r$  bits that corresponds to the information bit series of  $k$  bits can be derived by the following expression (2):

[Expression 2]

$$p_i = \sum_{j=1}^k a_{ij} c_j = a_{i1} c_1 + a_{i2} c_2 + \dots + a_{ik} c_k \quad (2)$$

[0007] Here,  $a_{ij}$  in the expression (2) represents the  $(i, j)$  element of the  $r \times k$  matrix A (where  $i$  represents an integer between 1 and  $r$  inclusive, and  $j$  represents an integer between 1 and  $k$  inclusive). Therefore, in order to constitute an encoding apparatus of an error correction code, the  $r \times k$  matrix A is held in a storage unit such as a memory, and exclusive-OR operations need to be performed as many times as the number of "1"s in the elements of the matrix.

[0008] FIG. 7 shows an example of a conventional encoding apparatus relating to the low-density parity-check code. 51 in FIG. 7 is a redundant bit string calculating unit that performs the expression (2), 52 in FIG. 7 is a memory that holds the matrix A in the expression (1), and 53 in FIG. 7 is a switch.

[0009] In regard to the reduction of storage units and exclusive-OR units in an encoding apparatus, a method in which the amount of memory is reduced and exclusive-OR operations are simplified by limiting the check matrix to a matrix constituted by a cyclic permutation matrix in the block-matrix form and imposing regularity on the matrix A (for instance refer to Patent Document 1), and a method for constituting low-density parity-check codes in which the number of "1"s in the elements of the matrix A is minimized whereas the coding gain obtained by iterative decoding is maximized (for instance refer to Non-Patent Document 3) are known.

[0010] Further, as error correction codes with which an encoding apparatus can easily be realized, cyclic codes in which a redundant bit string is calculated by using only a polynomial dividing circuit are known; the cyclic codes are represented by Reed Solomon (RS) code and BCH code, in particular. Further, as with the aforementioned cyclic codes, an encoding apparatus can easily be realized by using convolutional codes.

[0011] However, in the cyclic codes or the convolutional codes having a long constraint length, the amount of calculation required for soft-decision decoding that performs decoding with a close-to-optimum precision is very large, offering a problem, and sufficient coding gain cannot be obtained, compared with the low-density parity-check codes in which decoding can easily be performed with a close-to-optimum precision using an iterative decoding unit utilizing the aforementioned sum-product algorithm, which is courted as a problem. As relatively simple codes that can be decoded with a close-to-optimum precision using an iterative decoding and used for an encoding apparatus, turbo codes (for instance, refer to Patent Document 2) are known, however, the turbo codes have low code rates (the ratio between the

lengths of the information bit string and the code bit string), therefore they are not suitable for a system that requires a high code rate.

[0012] The fact that the evaluation of the error-rate characteristics of the low-density parity-check codes and the performance prediction of the error-rate characteristics can be generally performed by a system called "density evolution" in an area where the error probability is sufficiently high is known (for instance, refer to Non-Patent Document 4). The performance prediction of the error-rate characteristics in an area where the error probability is low, particularly in an area called error floor, is evaluated by an experimental method utilizing a computer simulation.

[0013] As described above, the conventional encoding apparatus relating to a low-density parity-check code are implemented by the storage units that hold the matrix A in the expression (1) and the operation processing units that perform the operation of the expression (2). Further, the evaluation of the error-rate characteristics is performed experimentally.

[Patent Document 1]

[0014] Japanese Patent Kokai Publication No. JP-P2003-115768A (pp. 10-11, FIGS. 4 to 7)

[Patent Document 2]

[0015] U.S. Pat. No. 5,446,747 (p. 2, FIG. 1)

[Non-Patent Document 1]

[0016] Gallager, R., "Low-Density Parity-Check Codes," MIT Press, 1963

[Non-Patent Document 2]

[0017] MacKny, D. J. C., "Good Error-Correcting Codes Based on Very Sparse Matrices," IEEE Transactions on Information Theory, Vol. 45, No. 2, March 1999, pp. 399-431

[Non-Patent Document 3]

[0018] Richardson, T. J., and Urbanke, R. L., "Efficient Encoding of Low-Density Parity-Check Codes," IEEE Transactions on Information Theory, Vol. 47, No. 2, September 2001, pp. 638-656

[Non-Patent Document 4]

[0019] Richardson, T. J., Shokrollahi, M. A., and Urbanke, R. L., "Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes," IEEE Transactions on Information Theory, Vol. 47, No. 2, September 2001, pp. 619-637

## DISCLOSURE OF THE INVENTION

Problems to be Solved by the Invention

[0020] Since the conventional error correction encoding apparatus relating to the low-density parity-check code described above is implemented by the storage units that hold the matrix A in the expression (1) and the operation processing units that perform the operation of the expression (2), the size of the encoding apparatus is very large compared with the apparatuses utilizing cyclic codes such as Reed Solomon code or convolutional codes. Particularly, under circumstances in which the demands for the device size and power consumption are high such as satellite

communications and mobile telecommunications, the amount of the storage units and the exclusive-OR units needs to be further reduced.

[0021] Further, while an encoding apparatus can be realized with relative ease using the turbo codes, it is difficult to apply such an apparatus to systems that require high code rates since the turbo code have low code rates. Because of the problems described above, in order to obtain large coding gain using the low-density parity-check codes, a large amount of calculation is required for encoding process, and the configuration of the apparatus become complex, particularly in communication systems in which high code rates are demanded.

[0022] Further, in the conventional error correction encoding apparatus, the evaluation of the error-rate characteristics has to be performed experimentally. The prediction of the error-rate characteristics in an area where the error probability is low, particularly in an area called "error floor", or the prediction of the error probability, in which the error floor is observed, is an important issue in evaluating the reliability of a communication system. An experimental method utilizing a computer simulation is effective, however, it is time-consuming to experimentally simulate the characteristics of an area with an error probability of  $10^{-12}$  considering the capability of current computers.

[0023] Accordingly, it is an object of the present invention to solve the problems above, and provide an error correction encoding apparatus wherein the apparatus structure is simple, an iterative decoding is used to achieve decoding with a close-to-optimum precision, and a simple mathematical expression is used to perform evaluation of the characteristic of an error floor area without using any computer experiments, and an error correction encoding method used therein.

Means to Solve the Problems

[0024] An error correction encoding apparatus according to the present invention is an error correction encoding apparatus that uses a low-density parity-check code and comprises  $(m-1)$  polynomial multiplying units (where  $m$  represents an integer equal to or greater than two) that respectively receive a block having a length  $n$  (where  $n$  represents an integer equal to or greater than two) of an information bit string divided into the  $(m-1)$  blocks of bit strings having the length  $n$  and a single block of a bit string having a length  $(n-r)$  (where  $r$  represents an integer between 1 and  $n$  inclusive), perform polynomial multiplication, and respectively output a bit series having the length  $n$ ; an adder unit that adds each output of the  $(m-1)$  polynomial multiplying units; and a polynomial dividing unit that performs polynomial division of an output result of the adder unit and the block having the length  $(n-r)$  and outputs a redundant bit series having a length  $r$ .

[0025] An error correction encoding method according to the present invention is an error correction encoding method in which a low-density parity-check code is used;  $(m-1)$  polynomial multiplying units (where  $m$  represents an integer equal to or greater than two) respectively receive a block having a length  $n$  (where  $n$  represents an integer equal to or greater than two) of an information bit string divided into the  $(m-1)$  blocks of bit strings having the length  $n$  and a single block of a bit string having a length  $(n-r)$  (where  $r$  represents

an integer between 1 and  $n$  inclusive), perform polynomial multiplication, and respectively output a bit series having the length  $n$ ; an adder unit adds each output of the  $(m-1)$  polynomial multiplying units; and a polynomial dividing unit performs polynomial division of an output result of the adder unit and the block having the length  $(n-r)$  and outputs a redundant bit series having a length  $r$ .

[0026] In other words, in order to achieve the object described above, the error correction encoding apparatus according to the present invention comprises  $(m-1)$  polynomial multiplying units that further divide an information bit string having a length  $K$  (where  $K$  represents an integer), which has been blocked for an error correction encoding, into  $(m-1)$  blocks having a length  $n$  and a single block having a length  $(n-r)$  (where  $m$  and  $n$  represent integers equal to or greater than two and  $r$  represents an integer between 1 and  $n$  inclusive), receive  $(m-1)$  blocks, which have the length  $n$ , among divided information bit strings, perform polynomial multiplication, and output a series having the length  $n$ ; an adder unit that adds each output of the  $(m-1)$  polynomial multiplying units; and a polynomial dividing unit that receives an output result of the adder unit and the block having the length  $(n-r)$ , performs polynomial division, and outputs a redundant bit series having a length  $r$ .

[0027] By having the configuration described above, the error correction encoding apparatus of the present invention can be realized with a simple apparatus structure since it is formed of polynomial multiplying units and a polynomial dividing unit, and the amount of calculation required for encoding processing and the apparatus scale can be reduced. Further, the number of codewords with minimum weight in the error correction encoding apparatus of the present invention can be made small by the selection of the wiring connection in the polynomial multiplying units and wiring connection in the polynomial dividing unit.

[0028] Therefore, in the error correction encoding apparatus of the present invention, the apparatus scale is small, the apparatus structure is simple, and a large coding gain can be obtained using an iterative decoding method, contributing to improvement in the reliability and a reduction of the required power in communication systems.

[0029] Further, since accurate approximations of the minimum distance and the number of codewords with the minimum weight are made possible by the way selections are made for the wiring connection in the polynomial multiplying units and the polynomial dividing unit in the error correction encoding apparatus of the present invention, it is possible to easily calculate good approximations of the error-rate characteristics, particularly the error probability in an error floor area, in a typical communication system to which the present invention is applied. Therefore, it becomes possible to quantitatively evaluate the reliability of the communication system even when it is too time-consuming to perform an experimental evaluation using a computer simulation or when the amount of calculation is too much to do so.

#### Meritorious Effects of the Invention

[0030] In the present invention, by having the configuration and operation described below, the apparatus structure can be made simple, decoding with a close-to-optimum precision can be achieved using an iterative decoding, and

evaluation of the characteristic of an error floor area can be performed using a simple mathematical expression without using any computer experiments.

#### BRIEF DESCRIPTION OF THE DRAWINGS

[0031] FIG. 1 is a block diagram showing the configuration of an error correction encoding apparatus according to an example of the present invention.

[0032] FIG. 2 is a block diagram showing the configuration of an  $(n-1)$ -th-order polynomial multiplying unit shown in FIG. 1.

[0033] FIG. 3 is a block diagram showing the configuration of an  $r$ -th-order polynomial dividing unit shown in FIG. 1.

[0034] FIG. 4 is a block diagram showing the detailed configuration of a polynomial multiplying block 1 shown in FIG. 1.

[0035] FIG. 5 is a flowchart showing an example of a method for calculating  $m$  polynomials that do not equal zero according to an example of the present invention.

[0036] FIG. 6 is a flowchart showing an example of a method for calculating polynomials according to another example of the present invention.

[0037] FIG. 7 is a block diagram showing an example of a conventional error correction encoding apparatus relating to the low-density parity-check code.

#### Explanations of Symbols

[0038] 1: polynomial multiplying block

[0039] 2:  $r$ -th-order polynomial dividing unit

[0040] 3, 4, 24, 34, 124: switch

[0041] 11, 40: serial/parallel converter unit

[0042] 12, 12-1 to 12- $(m-1)$ :  $(n-1)$ -th-order polynomial multiplying unit

[0043] 13: adder unit

[0044] 21-1 to 21- $r$ , 31-1 to 31- $n$ , 121-1 to 121- $n$ : register 22-1 to 22- $r$ , 32-1 to 32- $n$ , 122-1 to 122- $n$ : exclusive-OR unit

[0045] 23-1 to 23- $(r-1)$ , 33, 123-1 to 123- $n$ : switch indicating a wired or unwired state

#### PREFERRED MODES FOR CARRYING OUT THE INVENTION

[0046] Next, examples of the present invention will be described with reference to the drawings. FIG. 1 is a block diagram showing the configuration of an error correction encoding apparatus according to an example of the present invention. In FIG. 1, the error correction encoding apparatus according to an example of the present invention is formed by a polynomial multiplying block 1 comprising a serial/parallel (S/P) converter unit 11, an  $m-1$  number of  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$ , and an adder unit 13, a  $r$ -th-order polynomial dividing unit 2, and switches 3 and 4, and is an apparatus that converts an information bit string of  $(m-r)$  bits into a code bit string of  $nm$  bits (where  $m$  and  $n$  represent integers equal to or greater than two and  $r$  represents an integer from 1 to  $n$ , inclusive).

[0047] As described later, the polynomial multiplying block 1 is configured as shown in FIG. 4, however, in order to simplify the explanation, we will assume that the polynomial multiplying block 1 is constituted by an  $m-1$  number of  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$ .

[0048] An encoding method (or system) of the present invention is represented as a systematic encoding scheme in which  $(nm-r)$  bits from the first bit of the code bit string coincide with the information bit string while the remaining  $r$  bits become a redundant bit string for error correction.

[0049] The  $(m-1)$   $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$  further divide an information bit string having a length  $K$  (where  $K$  represents an integer), which has been blocked for an error correction encoding, into  $(m-1)$  blocks having a length  $n$  and a single block having a length  $(n-r)$  (where  $m$  and  $n$  represent integers equal to or greater than two and where  $r$  represents an integer from 1 to  $n$  inclusive), receive each of  $(m-1)$  blocks having the length  $n$  of (among) the divided information bit strings, performs  $(n-1)$ -th-order polynomial multiplication, and each output a series, (or sequence) each having the length  $n$ .

[0050] The adder unit 13 adds up each output of the  $(m-1)$   $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$ . The  $r$ -th-order polynomial dividing unit 2 receives the block having the length  $(n-r)$  and an output result of the adder unit 13, performs  $r$ -th-order polynomial division, and outputs a redundant bit series having a length  $r$ .

[0051] FIG. 2 is a block diagram showing the configuration of the  $(n-1)$ -th-order polynomial multiplying unit. In FIG. 2, the  $(n-1)$ -th-order polynomial multiplying unit 12 is constituted by an  $n$  number of registers 121-1 to 121- $n$  and, an  $n$  number, at a maximum, of exclusive-OR units 122-1 to 122- $n$ . The  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$  in FIG. 1 are all configured identically to this  $(n-1)$ -th-order polynomial multiplying unit 12.

[0052] The  $(n-1)$ -th-order polynomial multiplying unit 12 receives/outputs  $n$  bits, and sequentially receives an input bit string of  $n$  bits. After it has received all the bits, a switch 124 is switched over and the contents of the  $n$  registers 121-1 to 121- $n$  are sequentially output.

[0053] In FIG. 2, 123-1 to 123- $n$  are switches that are wire-connected or wire-disconnected by a predetermined bit string of  $n$  bits  $(h_1, h_2, \dots, h_n)$ . When  $h_i$  is 1, a part marked  $h_i$  is wire-connected, and when  $h_i$  is 0, the part marked  $h_i$  is wire-disconnected ( $i$  is an integer from 1 to  $n$  inclusive). How this bit string of  $n$  bits  $(h_1, h_2, \dots, h_n)$  is selected will be described later.

[0054] FIG. 3 is a block diagram showing the configuration of the  $r$ -th-order polynomial dividing unit shown in FIG. 1. In FIG. 3, the  $r$ -th-order polynomial dividing unit 2 is constituted by an  $r$  number of registers 21-1 to 21- $r$ , an  $r$  number, at a maximum, of exclusive-OR units 22-1 to 22- $r$ , and a switch 24.

[0055] The  $r$ -th-order polynomial dividing unit 2 receives the resultant  $n$  bits of exclusive-OR operations between  $(n-r)$  information bits and each output bit of the  $(m-1)$   $(n-1)$ -th-order polynomial multiplying units 121-1 to 121- $n$ , and outputs  $r$  bits. After it has received the  $(n-r)$  information bits, the  $r$ -th-order polynomial dividing unit 2 switches-over a switch 24 and sequentially outputs the results of exclusive-OR operations between the remaining  $r$  bits of each output of the  $(n-1)$ -th-order polynomial multiplying units 121-1 to

121- $(m-1)$  and the contents of the  $r$  registers 21-1 to 21- $r$  in FIG. 3 (at this time, the input of the information bit string is set to 0).

[0056] The output  $r$  bits of the  $r$ -th-order polynomial dividing unit 2 become the redundant bit string of  $r$  bits for the information bit string of  $(nm-r)$  bits. In FIG. 3, 23-1 to 23- $(r-1)$  are switches that are wire-connected or disconnected by a predetermined bit string of  $(r-1)$  bits  $(u_1, u_2, \dots, u_{r-1})$ . When  $u_i$  is 1, a part marked  $u_i$  is wire-connected, and when  $u_i$  is 0, the part marked  $u_i$  is disconnected ( $i$  is an integer from 1 to  $(r-1)$ , inclusive). How this bit string of  $(r-1)$  bits  $(u_1, u_2, \dots, u_{r-1})$  is selected will be described later.

[0057] FIG. 4 is a block diagram showing the detailed configuration of the polynomial multiplying block 1 shown in FIG. 1. In FIG. 4, registers are shared. 33 in FIG. 4 represents a switch that is connected or disconnected by a predetermined bit string of  $n(m-1)$  bits. When the same method is applied to selection of this bit string of  $n(m-1)$  bits and the bit string that determines the wire-connection state of the switches in the  $(n-1)$ -th-order polynomial multiplying units 12 in FIG. 2, the relationship between an input and an output shown in FIG. 4 coincides with the relationship between the input and the output of the polynomial multiplying block 1 realized by using a  $(m-1)$  number of  $(n-1)$ -th-order polynomial multiplying units 12 in FIG. 2.

[0058] A check matrix that corresponds to the error correction encoding apparatus according to an example of the present invention shown in FIG. 1 is indicated in the following expression:

[Expression 3]

$$H = (H_1 | H_2 | \dots | H_m) \quad (3)$$

[0059] The check matrix in the expression (3) is a linear arrangement of an  $m$  number of  $n \times n$  cyclic matrices, and  $H_i$  in the expression (3) indicates the  $n \times n$  cyclic matrix (where  $i$  is an integer from 1 to  $m$ , inclusive). The cyclic matrix is expressed as follows.

[Expression 4]

$$H_i = \begin{bmatrix} f_i^{(0)} & f_i^{(1)} & \dots & f_i^{(n-1)} \\ f_i^{(1)} & f_i^{(2)} & \dots & f_i^{(0)} \\ \vdots & \vdots & \ddots & \vdots \\ f_i^{(n-1)} & f_i^{(0)} & \dots & f_i^{(n-2)} \end{bmatrix} \quad (4)$$

In the expression (4), the positions of the first row vectors are shifted by one bit to the left in the second row, and the positions of the first row vectors are shifted by  $(k-1)$  bits to the left in a  $k$ -th row thereafter (where  $k$  represents an integer from two to  $n$ , inclusive).

[0060] The first row vectors of the  $n \times n$  cyclic matrix in the expression (4) is expressed by the following expression:

[Expression 5]

$$f_i^{(0)}(x) = f_i^{(0)} + f_i^{(1)}x + \dots + f_i^{(n-1)}x^{n-1} = \sum_{j=0}^{n-1} f_i^{(j)}x^j \quad (5)$$

In the expression (5), the first row vectors of the  $n \times n$  cyclic matrix are expressed by a polynomial of an order of  $(n-1)$  or less as  $f^{(i)}(x)$  (where  $i$  is an integer from 1 to  $m$ , inclusive).

[0061] The selection of the  $n$  bit string that determines the wiring-connection between the  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$  and the  $r$ -th-order polynomial dividing unit 2 is determined by the selection of  $m$  polynomials of an order of  $(n-1)$  or less:  $f^{(1)}(x), f^{(2)}(x), \dots, f^{(m)}(x)$ . First, a method for selecting the  $m$  polynomials of the order of  $(n-1)$  or less:  $f^{(1)}(x), f^{(2)}(x), \dots, f^{(m)}(x)$  will be described.

[0062] Regarding the polynomial  $f(x)$  of the order of  $(n-1)$  or less, a set  $s(f(x))$  is defined by the following expression:

[Expression 6]

$$s(f(x)) = \{i | f_i = 0, 0 \leq i < n\} \quad (6)$$

Here, the coefficient of the term of the order  $i$  of  $f(x)$  is indicated as  $f_i$ . The set  $s(f(x))$  is a subset of  $n$  integers from 0 to  $(n-1)$ , inclusive, determined by the polynomial  $f(x)$ .

[0063] Further, for integers  $v$  from 1 to  $(n-1)$ , inclusive, a subset  $\lambda_v(f(x))$  of a Cartesian product  $s(f(x)) \times s(f(x))$  is defined by the following expression:

[Expression 7]

$$\lambda_v(f(x)) = \{(i, j) | j-v \bmod n, i \in s(f(x)), j \in s(f(x))\} \quad (7)$$

[0064] One of the conditions for selecting the  $m$  polynomials of the orders not more than  $(n-1)$ :  $f^{(1)}(x), f^{(2)}(x), \dots, f^{(m)}(x)$  is that the  $m$  polynomials satisfy the following expression for all the integers  $v$  between 1 and  $(n-1)$  inclusive:

[Expression 8]

$$|\lambda_v(f^{(1)}(x))| + |\lambda_v(f^{(2)}(x))| + \dots + |\lambda_v(f^{(m)}(x))| = \sum_{i=1}^m |f^{(i)}(x)| \leq 1 \quad (8)$$

[0065] Here, the number of elements in a set  $A$  will be indicated as  $|A|$ . A second condition for selecting the  $m$  polynomials of the orders not more than  $(n-1)$ :  $f^{(1)}(x), f^{(2)}(x), \dots, f^{(m)}(x)$  is that each of  $(m-1)$  polynomials  $f^{(2)}(x), \dots, f^{(m)}(x)$  is divisible by the polynomial  $f^{(1)}(x)$  with the polynomial  $(x^{n-1})$  as a divisor. Further, these  $(m-1)$  number of quotient polynomials are expressed as  $g^{(2)}(x), g^{(3)}(x), \dots, g^{(m)}(x)$  as indicated by the following expression:

[Expression 9]

$$f^{(i)}(x) = g^{(i)}(x) f^{(1)}(x) \bmod (x^{n-1}), \quad i=2, 3, 4, \dots, m \quad (9)$$

[0066] The first condition [the expression (8)] for selecting the  $m$  polynomials of the orders not more than  $(n-1)$ :  $f^{(1)}(x), f^{(2)}(x), \dots, f^{(m)}(x)$  is a necessary condition in order to achieve a decoding with a close-to-optimum precision using an iterative decoding of the low-density parity-check code, typically, the sum-product algorithm etc. As a matter of fact, by satisfying this condition [the expression (8)], a maximum number of 1s included in each row vector of the check matrix shown in the expression (3) becomes  $(nm)^{1/2}$ , and the check matrix becomes a sparse matrix.

[0067] The second condition [the expression (9)] is a necessary condition in order to have the error correction encoding apparatus shown in FIG. 1 accurately calculate a redundant bit string. Examples of polynomials that satisfy the both conditions [the expressions (8) and (9)] will be described later.

[0068] Next, the wiring connection in the  $(n-1)$ -th-order polynomial multiplying unit 12 shown in FIG. 2 will be described. As mentioned above, the wiring connection of the switches 123-1 to 123- $n$  in the  $(n-1)$ -th-order polynomial multiplying unit 12 in FIG. 2 is determined by a predetermined bit string of  $n$  bits  $(h_1, h_2, \dots, h_n)$ . When  $h_i$  is 1, a part marked  $h_i$  is connected, and when  $h_i$  is 0, the part marked  $h_i$  is disconnected ( $j$  is an integer between 1 and  $m$  inclusive).

[0069] This bit string of  $n$  bits  $(h_1, h_2, \dots, h_n)$  is selected as follows. As mentioned earlier, the error correction encoding apparatus comprises an  $(m-1)$  number of  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$ , and an  $n$  bit string that determines the wiring connection of the  $i$ -th  $(n-1)$ -th-order polynomial multiplying units 12- $i$  is expressed as follows.

[Expression 10]

$$h_1^{(i)}, h_2^{(i)}, \dots, h_n^{(i)} \quad (10)$$

(where  $i$  represents an integer between 1 and  $(m-1)$  inclusive). This  $n$  bit string is defined by the following expression (11) using  $(m-1)$  polynomials  $g^{(2)}(x), g^{(3)}(x), g^{(m)}(x)$  indicated in the expression (9):

[Expression 11]

$$\begin{cases} h_1^{(i)} = g_1^{(i+1)-1} \\ h_j^{(i)} = g_{j-1}^{(i+1)-1}, \quad j=2, 3, \dots, n \end{cases} \quad (11)$$

[Expression 12]

$$g^{(k)}(x) = \sum_{j=1}^n g_j^{(k)} x^{j-1} = g_1^{(k)} + g_2^{(k)} x + g_3^{(k)} x^2 + \dots + g_n^{(k)} x^{n-1} \quad (12)$$

(where  $k$  is an integer between two and  $m$  inclusive). Further, the bit string of  $n(m-1)$  bits that determines the wiring connection of the switches in FIG. 4 is similarly defined by the expression (10).

[Expression 13]

$$h_1^{(i)}, h_2^{(i)}, \dots, h_n^{(i)}, \quad i=1, 2, \dots, m-1 \quad (13)$$

[0070] Next, the wiring connection in the  $r$ -th-order polynomial dividing unit 2 shown in FIG. 3 will be described. The switches 23-1 to 23- $(r-1)$  in FIG. 3 are wire-connected or wire-disconnected by a predetermined bit string of  $(r-1)$  bits  $(u_1, u_2, \dots, u_{r-1})$ . When  $u_i$  is 1, a part marked  $u_i$  is connected, and when  $u_i$  is 0, the part marked  $u_i$  is disconnected ( $j$  is an integer between 1 and  $(r-1)$  inclusive). The bit string of  $(r-1)$  bits  $(u_1, u_2, \dots, u_{r-1})$  is defined by the following expression using the aforementioned polynomial  $f^{(1)}(x)$ :

[Expression 14]

$$\frac{1+u_{r-1}x+\dots+u_2x^{r-2}+\dots+u_1x^{r-1}+x^r-1}{x^{r-1}} \bmod f^{(1)}(x) \quad (14)$$

[0071] Here,  $\text{gcd}^{(r)}(x)$ ,  $x^{n-1}$  represents the greatest common polynomial of  $f^{(r)}(x)$  and  $(x^{n-1})$ , and the order of this greatest common polynomial is  $(n-r)$ . As mentioned earlier,  $r$  is the redundant bit number of the encoding apparatus of the present invention. In other words, the information bit number of the error correction encoding apparatus of the present invention is the bit number obtained by, adding the order number of the greatest common polynomial of  $f^{(r)}(x)$  and  $(x^{n-1})$  to  $n(m-1)$ .

[0072] Next, the operation of the present example will be described. The error correction encoding apparatus shown in FIG. 1 sequentially receives the information bit string having the length  $K=(nm-r)$  bit (where  $m$  and  $n$  represent integers equal to or greater than two, and  $r$  represents an integer between 1 and  $n$  inclusive), which has been blocked for an error correction encoding. The information bit series is divided into a first block including the first bit to an  $n(m-1)$ -th bit and a second block including an  $n(m-1)+1$ -th bit to an  $(nm-r)$ -th bit.

[0073] The polynomial multiplying block 1 constituted by the  $(m-1)$ -th  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$  sequentially receives the first block, and after the switch 3 has been switched over, the second block is sequentially received by the  $r$ -th-order polynomial dividing unit 2. The first block having a length  $n(m-1)$  in the information bit string received by the polynomial multiplying block 1 is converted by the serial/parallel converter unit 11 into  $(m-1)$  bits, and each bit of the converted  $(m-1)$  bits is sequentially supplied to the  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$ .

[0074] There are an  $(m-1)$  number of the  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$ , and an  $i$ -th bit (where  $i$  represents an integer between 1 and  $(m-1)$  inclusive) of the serial/parallel converted  $(m-1)$  bits is supplied to the  $i$ -th  $(n-1)$ -th-order polynomial multiplying units 12-1. In other words, the first block is further divided into  $(m-1)$  blocks having the length  $n$  by the serial/parallel converter unit 11, and each of the  $(m-1)$  blocks having the length  $n$  is received by each of the  $(m-1)$   $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$ .

[0075] Each of the  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$  receives the divided  $n$  bits, and their output bit number is  $n$  bit. The bitwise exclusive-OR of each output of the  $(m-1)$   $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$  becomes the  $n$ -bit output of the polynomial multiplying block 1.

[0076] Next, the operation of the  $(n-1)$ -th-order polynomial multiplying unit 12 shown in FIG. 2 will be described. The contents of the registers 121-1 to 121- $n$  are all initialized to zero, and they sequentially receive the  $n$ -bit string bit by bit. During this time, the switch 124 is set so that there will be feedback. After all the  $n$  bits have been received, the switch 124 is switched, and the contents of the registers 121-1 to 121- $n$  are sequentially outputted.

[0077] The error correction encoding apparatus shown in FIG. 1 uses the  $(m-1)$  number of  $(n-1)$ -th-order polynomial multiplying units 12. Since the exclusive-OR of each output of the  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$  becomes the output of the polynomial multiplying block 1, each of the registers (121-1 to 121- $n$ ) can be shared with the other  $(n-1)$ -th-order polynomial multiplying units 12-1 to 12- $(m-1)$ .

[0078] The configuration of the polynomial multiplying block 1 shown in FIG. 4 can be achieved as described above. The operation in the configuration in FIG. 4 is the same as described except for the fact that the registers 31-1 to 31- $n$  are shared, and a desired output result can be obtained with this configuration shown in FIG. 4.

[0079] Next, the  $r$ -th-order polynomial dividing unit 2 shown in FIG. 3 will be described. The contents of the registers 21-1 to 21- $r$  are all initialized to zero, and they sequentially and simultaneously receive the last half  $(n-r)$  bits of the information bit string and the  $n$ -bit output of the polynomial multiplying block 1 bit by bit. During this time, the switch 24 in FIG. 3 is set so that there will be feedback. After the  $(n-r)$ -bit information bit string has been received, the switch 24 is switched, and the exclusive-ORs of the remaining  $r$  bits of the output of the polynomial multiplying block 1 and the content of each register (21-1 to 21- $r$ ) are sequentially outputted (the input of the information bit string is set to zero at this time).

[0080] The  $r$ -bit output of the  $r$ -th-order polynomial dividing unit 2 shown in FIG. 3 becomes the redundant bit string for the information bit string of  $(nm-r)$  bits. When the redundant bit number  $r$  coincides with  $n$ , it means that the bit number of the information bit string received by the  $r$ -th-order polynomial dividing unit 2 is zero, and in this case, the  $r$ -th-order polynomial dividing unit 2 simply outputs the  $n$ -bit output of the polynomial multiplying block 1.

[0081] Next, the output switch 4 of the error correction encoding apparatus shown in FIG. 1 will be described. The information bit string of  $(nm-r)$  bits becomes an output bit string of the error correction encoding apparatus simultaneously as it is received by the polynomial multiplying block 1 or the  $r$ -th-order polynomial dividing unit 2. At this time, if necessary, the first to the  $n(m-1)$ -th bits of the information bits received by the polynomial multiplying block 1 are rearranged so that their order coincides with the bit order indicated by the check matrix shown in the expression (3) before being outputted.

[0082] After all the  $(nm-r)$  bits of the information bit string have been outputted, the switch 4 is switched over so that the output of the  $r$ -th-order polynomial dividing unit 2 is active, and the  $r$ -bit output of the  $r$ -th-order polynomial dividing unit 2 is outputted as an output of the error correction encoding apparatus. As described, the present example is a systematic encoding apparatus in which the first to the  $n(m-r)$  bits of the code bit string coincide with the information bit string, and the remaining  $r$  bits become the redundant bit string for error correction.

[0083] By having such a configuration, an error correction encoding apparatus using a low-density parity-check code with a code rate equal to or greater than  $(n-1)/n$  can be constituted by a maximum of  $2n$  registers and  $2n$  exclusive-OR units in the present example.

[0084] Further, as described later, an error correction encoding apparatus wherein the coding gain is large and the error-rate characteristics in an error floor area can be calculated by a simple mathematical evaluation expression (or formula) can be constituted by the configuration of the present example.

[0085] Next, a concrete example for calculating polynomials that satisfy the aforementioned conditions (the expres-

sions (8) and (9)] will be described. Further, using this example, how a large coding gain can be obtained and how the error-rate characteristics in an error floor area can be calculated by a simple mathematical evaluation expression will be explained.

[0086]  $\alpha$  is a primitive element a finite field  $\text{GF}(2^{2^b})$  (where  $S$  is a positive integer), and  $R$  is a divisor of  $S$ , but is not 1 nor  $S$ . Further,  $n$  is represented by  $(2^S+1)(2^{R-1})$  and  $m$  by  $(2^S-2^{S-R})/(2^R-1)$ . A polynomial  $\psi^{(k)}(x)$  of an order not more than  $(n-1)$  is defined by the following expression:

[Expression 15]

$$\psi^{(k)}(x) = \sum_{j=0}^{n-1} x^{kj} \quad (15)$$

(where  $k$  represents integers between 0 and  $(2^S-1)/(2^R-1)-1$  inclusive.)

[0087] Here,  $L(K)$  is a subset of integers between 0 and  $(n-1)$  inclusive, and it is defined by the following expression:

[Expression 16]

$$L(K) = \{j | \text{Tr}_{2^{2^b}}(\alpha^{2^{2^b-j}}) = 1, 0 \leq j < n\} \quad (16)$$

Note that  $\text{Tr}_{2^{2^b}}$  in the expression (16) represents the trace from the finite field  $\text{GF}(2^{2^b})$  to  $\text{GF}(2^R)$ . There are  $m$  number of the polynomials  $\psi^{(k)}(x)$  that do not equal zero where  $k$  represents integers between 0 and  $(2^S-1)/(2^R-1)-1$  inclusive.

[0088] FIG. 5 is a flowchart showing an example of a method for calculating the  $m$  polynomials that do not equal zero. Whether or not the polynomial  $\psi^{(k)}(x)$  equals zero (where  $k$  represents integers between 0 and  $(2^S-1)/(2^R-1)-1$  inclusive) can be determined by whether or not the trace of  $\alpha^{2^{2^b-j}}$  (where  $\alpha$  is an element of a finite field  $\text{GF}(2^{2^b})$ ) to  $\text{GF}(2^R)$  equals zero. In other words, it can be determined by the following expression:

[Expression 17]

$$\text{Tr}_{2^{2^b}}(\alpha^{2^{2^b-j}}) = 0 \quad (17)$$

[0089] Since  $(2^S-1)/(2^R-1)$  multiplied by  $m$  is  $2^{2^b}-1$ , the set  $L(k)$  in the expression (16) corresponds to a set of elements that are a subset of the finite field  $\text{GF}(2^{2^b})$ , whose values coincide with each other when raised to the  $m$ -th power, and whose traces to the finite field  $\text{GF}(2^R)$  are 1. Further, even when the value of the trace above is not zero, the above expression works as long as the value of the trace is the same for each  $k$  (where  $k$  represents integers between 0 and  $(2^S-1)/(2^R-1)-1$  inclusive). Therefore, the polynomials  $\psi^{(k)}(x)$  are defined by a set of elements whose values coincide with each other when raised to the  $m$ -th power, and whose traces to the finite field  $\text{GF}(2^R)$  are a predetermined value other than zero and coincide with each other in a subset of the finite field  $\text{GF}(2^{2^b})$  (steps S1 to S7 in FIG. 5).

[0090] After the  $m$  polynomials, which are the output of the flowchart in FIG. 5, are properly arranged, they are represented as  $f^{(1)}(x)$ ,  $f^{(2)}(x)$ ,  $\dots$ ,  $f^{(m)}(x)$ . These  $m$  polynomials  $f^{(1)}(x)$ ,  $f^{(2)}(x)$ ,  $\dots$ ,  $f^{(m)}(x)$  satisfy the aforementioned conditions [the expressions (8) and (9)]. Here, these  $m$  polynomials should be arranged so that,

particularly, the polynomial  $f^{(1)}(x)$  is the minimal polynomial, which means that, among the polynomials  $\psi^{(k)}(x)$  that do not equal zero, the order number of its greatest common polynomial with  $(x^n-1)$  is the smallest.

[0091] The length  $N$  of the code bit string in the error correction encoding apparatus according to the present example constituted by the polynomials selected as described is as follows.

$$N = nm - (2^R+1)(2^S-2^{S-R})$$

The length  $K$  of the information bit string is not less than

$$n(m-1) - (2^R+1)(2^S-2^{S-R}-2^{S-1})$$

To be precise, as described above, it is actually the bit number obtained by adding the order number of the greatest common polynomial of  $f^{(1)}(x)$  and  $(x^n-1)$  to  $n(m-1)$ .

[0092] Further, a minimum distance  $d$  is represented by  $d=2^R+1$ , and the number  $A_d$  of the code bit strings with a weight  $d$  is at least  $(2^S+1)(2^S-2^{S-R})$ . In reality, the number  $A_d$  of the code bit strings with the weight  $d$  is very close to  $(2^S+1)(2^S-2^{S-R})$ , therefore it is a good approximation. In addition, the number of the code bit strings with the weight  $d$  is very small, and this fact has a good effect on the error-rate characteristics as described later.

[0093] Using the approximations of the minimum distance  $d$  and the number of the code bit strings with the weight  $d$ , an approximate value of a bit-error probability  $P_b$  when the modulation scheme is 2PSK (2 phase-shift keying) and the channel is the additive Gaussian channel can be calculated by the following expression:

[Expression 18]

$$P_b \sim \frac{d}{N} A_d Q \left( \sqrt{\frac{2d}{N} \frac{E_b}{N_0}} \right) \quad (18)$$

[0094] Here,  $d$  represents the minimum distance ( $d=2^R+1$ ),  $A_d$  the number of the code bit strings with the weight  $d$ ,  $N$  the length of the code bit string, and  $\gamma$  code rate ( $\gamma=N/N_0$ ). Further,  $E_b/N_0$  represents the SNR (signal-to-noise ratio) per bit in the additive Gaussian channel, and  $Q$  represents the Gaussian  $Q$  function, which can be expressed by the following expression:

[Expression 19]

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty \exp(-t^2/2) dt \quad (19)$$

[0095] By respectively substituting  $(2^R+1)$  and  $(2^S+1)(2^S-2^{S-R})$  for  $N$  and  $d$  and  $(2^S+1)(2^S-2^{S-R})$  for  $A_d$  in the expression (18), the approximate value of bit-error probability  $P_b$  can be calculated. This approximation is highly accurate in an error floor area, and it is effective for the evaluation of the characteristic of the error floor area when it is too time-consuming to perform an experimental evaluation using a computer simulation or when the amount of calculation is too much to do so.

[0096] Increasing the minimum distance  $d$  is effective for improving the error probability using the expression (18) and obtaining larger coding gain. There are  $m$  number of polynomials that do not equal zero among the polynomials  $\psi^{(k)}(x)$  ( $k=0, 1, \dots, (2^k-1)/(2^{k-1}-1)$ ) indicated by the expression (15). Instead of using all the  $m$  polynomials as described, it is possible to increase the minimum distance to approximately  $(2^k+1)$  by using only parts of the polynomials. This method will be described.

[0097]  $m'$  polynomials are selected from  $m$  polynomials  $\psi^{(k)}(x)$  (where  $k$  represents integers between 0 and  $(2^k-1)/(2^{k-1}-1)$ , inclusive that do not equal zero. After they have been properly arranged, assume that they are represented as  $f^{(1)}(x), f^{(2)}(x), \dots, f^{(m')}(x)$ . At this time, the polynomial  $\psi^{(k)}(x)$  and a polynomial  $\psi^{(k')}(x)$  should not be selected together at once. Here,  $r(k)$  represents the remainder when  $(2^k-1)$  is divided by  $(2^{k-1}-1)$ .

[0098] These  $m'$  polynomials  $f^{(1)}(x), f^{(2)}(x), \dots, f^{(m')}(x)$  satisfy the aforementioned conditions [the expressions (8) and (9)], and the length  $N$  of the code bit string in the encoding apparatus of the present invention constituted by the polynomials selected as described is  $N=m \cdot m'$ . The minimum distance is equal to or less than  $2^{k+1}$ , and the number of the code bit strings with a weight  $2^{k+1}$  is at least a value obtained by multiplying a binomial coefficient of  $m'$  and 2 by  $n$ . A value obtained by substituting a lower bound of the code bit strings with this weight  $2^{k+1}$  into the expression (18) can provide a very good approximation of the error-rate characteristics when the error probability is relatively low (when the signal-to-noise ratio of the channel is relatively high). Using this method, the code rate becomes lower compared with the case where all the  $m$  polynomials are used, however, the error characteristics are improved, instead.

[0099] The present example will be further described using exemplified concrete values. In the present example described above, we will assume  $S=6$  and  $R=3$ . In this case,  $n=455$  and  $m=8$ . Further, the polynomials  $\psi^{(k)}(x)$  of the expression (15) (where  $k$  represents integers between 0 and 8 inclusive) can be expressed as follows.

[Expression 20]

$$\begin{aligned} \psi^{(0)}(x) &= x^{124} + x^{175} + x^{279} + x^{254} + x^{281} + x^{338} \quad (20) \\ \psi^{(1)}(x) &= (\psi^{(0)}(x)) \bmod (x^{455} - 1) \\ \psi^{(2)}(x) &= x^{210} + x^{40} + x^{121} + x^{461} + x^{390} + x^{113} + x^{251} \\ &= g(x)\psi^{(0)}(x) \bmod (x^{455} - 1) \\ \psi^{(3)}(x) &= (\psi^{(0)}(x))^2 \bmod (x^{455} - 1) \\ \psi^{(4)}(x) &= (\psi^{(0)}(x))^3 \bmod (x^{455} - 1) \\ \psi^{(5)}(x) &= x^2 (\psi^{(0)}(x))^2 \bmod (x^{455} - 1) \\ \psi^{(6)}(x) &= (\psi^{(0)}(x))^2 \bmod (x^{455} - 1) \\ &= g(x)(\psi^{(0)}(x))^2 \bmod (x^{455} - 1) \\ \psi^{(7)}(x) &= x(g(x)\psi^{(0)}(x))^2 \bmod (x^{455} - 1) \\ \psi^{(8)}(x) &= (\psi^{(0)}(x))^2 \bmod (x^{455} - 1) \end{aligned}$$

Further,  $\psi^{(0)}(x)=0$ , and  $g(x)$  represents the quotient polynomial when  $\psi^{(k)}(x)$  is divided by  $\psi^{(1)}(x)$  using  $(x^{455}-1)$  as a divisor.

[0100] When the eight polynomials  $f^{(1)}(x), f^{(2)}(x), \dots, f^{(8)}(x)$  are represented by  $f^{(k)}(x)=\psi^{(k)}(x)$  (where  $k=1, 2, \dots$

$\dots, 8$ ), these eight polynomials satisfy the aforementioned conditions [the expressions (8) and (9)]. At this time, the length  $N$  of the code bit string is 3640 bits; the length  $K$  of the information bit string is 3288 bits; the code rate  $r$  is approximately 0.9; the minimum distance is 9; and the number  $A_9$  of the code bit strings with a weight 9 is equal to or more than 3640.

[0101] By substituting these values into the expression (18) and calculating the error probability, the fact that the signal-to-noise ratio per bit required for achieving a bit-error probability of  $10^{-12}$  after decoding is approximately 5 decibels can be found out. This can be achieved by utilizing an iterative decoding method.

[0102] A signal-to-noise ratio necessary to achieve an even lower bit-error rate using an iterative decoding method can also easily be calculated. Further, compared with the case where this code is encoded using the Reed Solomon code with the same code rate and decoded using a typical bounded distance decoding method, a coding gain of 2.0 decibels (dB) or more can be obtained when the bit-error probability after decoding is  $10^{-4}$ .

[0103] When four polynomials out of the eight polynomials  $\psi^{(1)}(x), \psi^{(2)}(x), \dots, \psi^{(8)}(x)$  are used and represented as  $f^{(1)}(x)=\psi^{(1)}(x), f^{(2)}(x)=\psi^{(2)}(x), f^{(3)}(x)=\psi^{(3)}(x), f^{(4)}(x)=\psi^{(4)}(x)$ , these four polynomials satisfy the aforementioned conditions [the expressions (8) and (9)] as well. At this time, the length  $N$  of the code bit string is 1820 bits; the length  $K$  of the information bit string is 1468 bits; the code rate  $r$  is approximately 0.8; the minimum distance is 16 or less; and the number  $A_{16}$  of the code bit strings with a weight 16 is 2730 or more.

[0104] By substituting these values into the expression (18) and calculating the error probability, the fact that the signal-to-noise ratio per bit required for achieving a bit-error probability of  $10^{-12}$  after decoding is approximately 4.4 decibels can be found out, and this can be achieved by utilizing an iterative decoding method. A signal-to-noise ratio necessary to achieve an even lower bit-error rate using an iterative decoding method can also easily be calculated. Further, compared with the case where this code is encoded using the Reed Solomon code with the same code rate and decoded using a typical bounded distance decoding method, a coding gain of 2.0 decibels (dB) or more can be obtained when the bit-error probability after decoding is  $10^{-4}$ .

[0105] FIG. 6 is a flowchart showing a method for calculating polynomials according to another example of the present invention. In FIG. 6,  $k$  represents a non-negative integer, and  $T$  represents a set of polynomial(s) of order(s) not more than  $(n-1)$  (where  $n$  represents an integer equal to or greater than two). Further,  $p$  represents powers of two such as 2 and 4. In an initial state,  $k=0$ ;  $T$  is an empty set; and polynomial(s) of order(s) not more than  $(n-1)$   $f(x)$  are selected randomly.

[0106] When these polynomials satisfy the aforementioned condition [the expression (8)] (in the expression (8),  $m$  is 1 and  $f^{(1)}(x)$  is  $f(x)$ ),  $f(x)$  are added to the elements of the set  $T$ , and the value of  $k$  is increased by one. In a  $k$ -th iterative processing in FIG. 6 (where  $k$  represents an integer equal to or greater than two), when the following polynomial (an expression (21)) calculated from the set of polynomials  $T$  and the polynomials  $f(x)$  described above satisfies the



aforementioned condition [the expression (8)] ( $m$  in the expression (8) is  $k$ ),  $f^{(k)}(x)$  is added to the elements of the set  $T$  and the value of  $k$  is increased by one.

[Expression 21]

$$f^{(k)}(x) = (f(x))^{2^{k-1} \bmod (2^m - 1)} \quad (21)$$

As soon as the number of polynomials included in the set of polynomials  $T$  reaches a predetermined number  $m$  (where  $m$  represents an integer equal to or greater than two),  $T$  is outputted and the processing ends (steps S11 to S17 in FIG. 6).

[0107] In FIG. 6, as described above, by the polynomials  $f(x)$ , which are randomly selected in the initial state, the following expression can be obtained:

[Expression 22]

$$f^{(k)}(x) = (f(x))^{2^{k-1} \bmod (2^m - 1)}, k=1, 2, \dots, m \quad (22)$$

Then the output  $T$  of the flowchart in FIG. 6 is  $T = \{f^{(1)}(x), f^{(2)}(x), \dots, f^{(m)}(x)\}$ , and they satisfy the aforementioned conditions [the expressions (8) and (9)]. Further, when the number of terms that do not equal zero in the polynomials  $f(x)$  selected in the initial state is represented by  $w$ , each of the polynomials included in  $T$  has  $w$  terms that do not equal zero.

[0108] The length  $N$  of the code bit string in an error correction encoding apparatus according to the present example constituted by the  $m$  polynomials  $f^{(1)}(x), f^{(2)}(x), \dots, f^{(m)}(x)$ , which are the output of the flowchart in FIG. 6, is  $N = nm$ , and the length  $K$  of the information bit string is at least  $n(m-1)$ . To be precise, as described above, it is actually the bit number obtained by adding the order number of the greatest common polynomial of  $f^{(1)}(x)$  and  $(x^m - 1)$  to  $n(m-1)$ .

[0109] When  $p$  is 2, the minimum distance  $d$  is  $d = w + 1$  and the number  $A_d$  of the code bit strings with a weight  $d$  is at least  $n(m-1)$ . In most of cases, a value obtained by substituting a lower bound of the number of the code bit strings with this minimum weight into the expression (18) can provide a very good approximation of the error-rate characteristics when the error probability is relatively low (when the signal-to-noise ratio of the channel is relatively high).

[0110] Further, when  $p$  is not 2, the number of the code bit strings with a weight  $2w$  is at least a value obtained by multiplying the binomial coefficient of  $m$  and 2 by  $n$ , and in most cases, a value obtained by substituting a lower bound of the number of the code bit strings with this minimum weight  $2w$  into the expression (18) can provide a very good approximation of the error-rate characteristics when the error probability is relatively low (when the signal-to-noise ratio of the channel is relatively high).

[0111] Next, the present example will be further described using concrete values as an example. When  $n=255$ ,  $m=4$ , and  $p=4$  in the present example, by  $f_2(x)$  expressed as below,

[Expression 23]

$$f_2(x) = x^{2^{255-1}} + x^{2^{255-2}} + x^{2^{255-4}} + x^{2^{255-8}} + x^{2^{255-16}} + x^{2^{255-32}} + x^{2^{255-64}} + x^{2^{255-128}} + x^{2^{255-256}} + x^{2^{255-512}} + x^{2^{255-1024}} + x^{2^{255-2048}} + x^{2^{255-4096}} + x^{2^{255-8192}} + x^{2^{255-16384}} + x^{2^{255-32768}} + x^{2^{255-65536}} + x^{2^{255-131072}} + x^{2^{255-262144}} + x^{2^{255-524288}} + x^{2^{255-1048576}} + x^{2^{255-2097152}} + x^{2^{255-4194304}} + x^{2^{255-8388608}} + x^{2^{255-16777216}} + x^{2^{255-33554432}} + x^{2^{255-67108864}} + x^{2^{255-134217728}} + x^{2^{255-268435456}} + x^{2^{255-536870912}} + x^{2^{255-1073741824}} + x^{2^{255-2147483648}} + x^{2^{255-4294967296}} + x^{2^{255-8589934592}} + x^{2^{255-17179869184}} + x^{2^{255-34359738368}} + x^{2^{255-68719476736}} + x^{2^{255-137438953472}} + x^{2^{255-274877906944}} + x^{2^{255-549755813888}} + x^{2^{255-1099511627776}} + x^{2^{255-2199023255552}} + x^{2^{255-4398046511104}} + x^{2^{255-8796093022208}} + x^{2^{255-17592186044416}} + x^{2^{255-35184372088832}} + x^{2^{255-70368744177664}} + x^{2^{255-140737488355328}} + x^{2^{255-281474976710656}} + x^{2^{255-562949953421312}} + x^{2^{255-1125899906842624}} + x^{2^{255-2251799813685248}} + x^{2^{255-4503599627370496}} + x^{2^{255-9007199254740992}} + x^{2^{255-18014398509481984}} + x^{2^{255-36028797018963968}} + x^{2^{255-72057594037927936}} + x^{2^{255-144115188075855872}} + x^{2^{255-288230376151711744}} + x^{2^{255-576460752303423488}} + x^{2^{255-1152921504606846976}} + x^{2^{255-2305843009213693952}} + x^{2^{255-4611686018427387904}} + x^{2^{255-9223372036854775808}} + x^{2^{255-18446744073709551616}} + x^{2^{255-36893488147419103232}} + x^{2^{255-73786976294838206464}} + x^{2^{255-147573952589676412928}} + x^{2^{255-295147905179352825856}} + x^{2^{255-590295810358705651712}} + x^{2^{255-1180591620717411303424}} + x^{2^{255-2361183241434822606848}} + x^{2^{255-4722366482869645213696}} + x^{2^{255-9444732965739290427392}} + x^{2^{255-18889465931478580854784}} + x^{2^{255-37778931862957161709568}} + x^{2^{255-75557863725914323419136}} + x^{2^{255-151115727451828646838272}} + x^{2^{255-302231454903657293676544}} + x^{2^{255-604462909807314587353088}} + x^{2^{255-1208925819614629174706176}} + x^{2^{255-2417851639229258349412352}} + x^{2^{255-4835703278458516698824704}} + x^{2^{255-9671406556917033397649408}} + x^{2^{255-19342813113834066795298816}} + x^{2^{255-38685626227668133590597632}} + x^{2^{255-77371252455336267181195264}} + x^{2^{255-154742504910672534362390528}} + x^{2^{255-309485009821345068724781056}} + x^{2^{255-618970019642690137449562112}} + x^{2^{255-1237940039285380274899124224}} + x^{2^{255-2475880078570760549798248448}} + x^{2^{255-4951760157141521099596496896}} + x^{2^{255-9903520314283042199192993792}} + x^{2^{255-19807040628566084398385987584}} + x^{2^{255-39614081257132168796771975168}} + x^{2^{255-79228162514264337593543950336}} + x^{2^{255-158456325028528675187087900672}} + x^{2^{255-316912650057057350374175801344}} + x^{2^{255-633825300114114700748351602688}} + x^{2^{255-1267650600228229401496703205376}} + x^{2^{255-2535301200456458802993406410752}} + x^{2^{255-5070602400912917605986812821504}} + x^{2^{255-10141204801825835211973625643008}} + x^{2^{255-20282409603651670423947251286016}} + x^{2^{255-40564819207303340847894502572032}} + x^{2^{255-81129638414606681695789005144064}} + x^{2^{255-162259276829213363391578010288128}} + x^{2^{255-324518553658426726783156020576256}} + x^{2^{255-649037107316853453566312041152512}} + x^{2^{255-1298074214633706907132624082305024}} + x^{2^{255-2596148429267413814265248164610048}} + x^{2^{255-5192296858534827628530496329220096}} + x^{2^{255-10384593717069655257060992658440192}} + x^{2^{255-20769187434139310514121985316880384}} + x^{2^{255-41538374868278621028243970633760768}} + x^{2^{255-83076749736557242056487941267521536}} + x^{2^{255-166153499473114484112974882535043072}} + x^{2^{255-332306998946228968225949765070086144}} + x^{2^{255-664613997892457936451899530140172288}} + x^{2^{255-1329227995784915872903799060280344576}} + x^{2^{255-2658455991569831745807598120560689152}} + x^{2^{255-5316911983139663491615196241121378304}} + x^{2^{255-10633823966279326983230392482242756608}} + x^{2^{255-21267647932558653966460784964485513216}} + x^{2^{255-42535295865117307932921569928971026432}} + x^{2^{255-85070591730234615865843139857942052864}} + x^{2^{255-170141183460469231731686279715884105728}} + x^{2^{255-340282366920938463463372559431768211456}} + x^{2^{255-680564733841876926926745118863536422912}} + x^{2^{255-1361129467683753853853490237727072845824}} + x^{2^{255-2722258935367507707706980475454145691648}} + x^{2^{255-5444517870735015415413960950908291383296}} + x^{2^{255-10889035741470030830827921901816582766592}} + x^{2^{255-21778071482940061661655843803633165533184}} + x^{2^{255-43556142965880123323311687607266331066368}} + x^{2^{255-87112285931760246646623375214532662132736}} + x^{2^{255-174224571863520493293246750429065324265472}} + x^{2^{255-348449143727040986586493500858130648530944}} + x^{2^{255-696898287454081973172987001716261297061888}} + x^{2^{255-1393796574908163946345974003432522594123776}} + x^{2^{255-2787593149816327892691948006865045188247552}} + x^{2^{255-5575186299632655785383896013730090376495104}} + x^{2^{255-11150372599265311570767792027460180752990208}} + x^{2^{255-22300745198530623141535584054920361505980416}} + x^{2^{255-44601490397061246283071168109840723011960832}} + x^{2^{255-89202980794122492566142336219681446023921664}} + x^{2^{255-178405961588244985132284672439362892047843328}} + x^{2^{255-356811923176489970264569344878725784095686656}} + x^{2^{255-713623846352979940529138689757451568191373312}} + x^{2^{255-1427247692705959881058277379514903136382746624}} + x^{2^{255-2854495385411919762116554759029806272765493248}} + x^{2^{255-5708990770823839524233109518059612545530986496}} + x^{2^{255-11417981541647679048466219036119225091061972992}} + x^{2^{255-22835963083295358096932438072238450182123945984}} + x^{2^{255-45671926166590716193864876144476900364247891968}} + x^{2^{255-91343852333181432387729752288953800728495783936}} + x^{2^{255-182687704666362864775459504577907601456991567872}} + x^{2^{255-365375409332725729550919009155815202913983735744}} + x^{2^{255-730750818665451459101838018311630405827967471488}} + x^{2^{255-1461501637330902918203676036623260811655934942976}} + x^{2^{255-2923003274661805836407352073246521623111869885952}} + x^{2^{255-5846006549323611672814704146493043246223739771904}} + x^{2^{255-11692013098647223345629408292986086492447479543808}} + x^{2^{255-23384026197294446691258816585972172984894959087616}} + x^{2^{255-46768052394588893382517633171944345969789918175232}} + x^{2^{255-93536104789177786765035266343888691939579836350464}} + x^{2^{255-187072209578355573530070532687777383879159672700928}} + x^{2^{255-374144419156711147060141065375554767758319345401856}} + x^{2^{255-748288838313422294120282130751109535516638690803712}} + x^{2^{255-1496577676626844588240564261502219071033277381607424}} + x^{2^{255-2993155353253689176481128523004438142066554763214848}} + x^{2^{255-5986310706507378352962257046008876284133109526429696}} + x^{2^{255-11972621413014756705924514092017752568266219052859392}} + x^{2^{255-23945242826029513411849028184035505136532438105718784}} + x^{2^{255-47890485652059026823698056368071010273064876211437568}} + x^{2^{255-95780971304118053647396112736142020546129752422875136}} + x^{2^{255-191561942608236107294792225472284041092259504845750272}} + x^{2^{255-383123885216472214589584450944568082184519009691500544}} + x^{2^{255-766247770432944429179168901889136164369038019383001088}} + x^{2^{255-1532495540865888858358337803778272328738076038766002176}} + x^{2^{255-3064991081731777716716675607556544657476152077532004352}} + x^{2^{255-6129982163463555433433351215113089314952304155064008704}} + x^{2^{255-12259964326927110866866702430226178629904608310128017408}} + x^{2^{255-24519928653854221733733404860452357259809216620256034816}} + x^{2^{255-49039857307708443467466809720904714519618433240512069632}} + x^{2^{255-98079714615416886934933619441809429039236866481024139264}} + x^{2^{255-196159429230833773869867238883618858078473732962048278528}} + x^{2^{255-392318858461667547739734477767237716156947465924096557056}} + x^{2^{255-784637716923335095479468955534475432313894931848193114112}} + x^{2^{255-1569275433846670190958937911068950864627789863696386328224}} + x^{2^{255-3138550867693340381917875822137901729255579727392772656448}} + x^{2^{255-6277101735386680763835751644275803458511159454785545312896}} + x^{2^{255-12554203470773361527671503288551606917022318909571090725792}} + x^{2^{255-25108406941546723055343006577103213834044637819142181451584}} + x^{2^{255-50216813883093446110686013154206427668089275638284362903168}} + x^{2^{255-100433627766186892221372026308412855336178551276568725806336}} + x^{2^{255-200867255532373784442744052616825710672357102553137451612672}} + x^{2^{255-401734511064747568885488105233651421344714205106274903325344}} + x^{2^{255-803469022129495137770976210467302842689428410212549806650688}} + x^{2^{255-1606938044258990275541952420934605685378856820425099613301376}} + x^{2^{255-3213876088517980551083904841869211370757713640850199226602752}} + x^{2^{255-6427752177035961102167809683738422741515427281700398453205504}} + x^{2^{255-12855504354071922204335619367476845483030854563400796906411008}} + x^{2^{255-25711008708143844408671238734953690966061709126801593812822016}} + x^{2^{255-51422017416287688817342477469907381932123418253603187625644032}} + x^{2^{255-102844034832575377634684954939814763864246836507206375251288064}} + x^{2^{255-205688069665150755269369909879629527728493673014412750502576128}} + x^{2^{255-411376139330301510538739819759259055456987346028825501005152256}} + x^{2^{255-822752278660603021077479639518518110913974692057651002010304512}} + x^{2^{255-1645504557321206042154959279037036221827949384115302004020609024}} + x^{2^{255-3291009114642412084309918558074072443655898768230604008041218048}} + x^{2^{255-6582018229284824168619837116148144887311797536461208016082436096}} + x^{2^{255-13164036458569648337239674232296289774623595072922416032164872192}} + x^{2^{255-26328072917139296674479348464592579549247190145844832064329744384}} + x^{2^{255-52656145834278593348958696929185159098494380291689664128659488768}} + x^{2^{255-105312291668557186697917393858370318196988760583379328257318977536}} + x^{2^{255-210624583337114373395834787716740636393977521166758656514637955072}} + x^{2^{255-421249166674228746791669575433481272787955042333517313029275910144}} + x^{2^{255-842498333348457493583339150866962545575910084667034626058551820288}} + x^{2^{255-1684996666696914987166678301733925091151820169334069252117103640576}} + x^{2^{255-3369993333393829974333356603467850182303640338668138504234207281152}} + x^{2^{255-6739986666787659948666713206935700364607280677336277008468414562304}} + x^{2^{255-13479973333575319897333426413871400729214561354672554016936829124608}} + x^{2^{255-26959946667150639794666852827742801458429122709345108033873658249216}} + x^{2^{255-53919893334301279589333705655485602916858245418690216067747316498432}} + x^{2^{255-107839786668602559178667411310971205833716490837380432135494632996864}} + x^{2^{255-215679573337205118357334822621942411667432981674760864270989265993728}} + x^{2^{255-431359146674410236714669645243884823334865963349521728541978531987456}} + x^{2^{255-862718293348820473429339290487769646669731926699043457083957063974912}} + x^{2^{255-1725436586697640946858678580975539293339463853398086914167914127949824}} + x^{2^{255-3450873173395281893717357161951078586678927706796173828335828255899648}} + x^{2^{255-6901746346790563787434714323902157173357855413592347656671656511799296}} + x^{2^{255-13803492693581127574869428647804314346715710827184695313343313023598592}} + x^{2^{255-27606985387162255149738857295608628693431421654369390626686626047197184}} + x^{2^{255-55213970774324510299477714591217$$

1. An error correction encoding apparatus using a low-density parity-check code, said apparatus comprising:

(m-1) polynomial multiplying units (where m represents an integer equal to or greater than two) that respectively receive a block having a length n (where n represents an integer equal to or greater than two) of an information bit string divided into said (m-1) blocks of bit strings having the length n and a single block of a bit string having a length (n-r) (where r represents an integer between 1 and n inclusive), perform polynomial multiplication, and respectively output a bit series having the length n;

an adder unit that adds each output of said (m-1) polynomial multiplying units; and

a polynomial dividing unit that performs polynomial division of an output result of said adder unit and said block having the length (n-r) and outputs a redundant bit series having a length r.

2. The error correction encoding apparatus as defined in claim 1, wherein said polynomial dividing unit and said polynomial multiplying units include a circuit in which registers and exclusive-OR circuits connected to outputs of said registers are connected in a cascade fashion in a plurality of stages; and output logics of said exclusive-OR circuits are set by wiring connection determined according to a predetermined polynomial operation so that output logics of said exclusive-OR circuits are non-inverted or inverted.

3. The error correction encoding apparatus as defined in claim 2, wherein a wiring polynomial that specifies the wiring connection in said polynomial dividing unit is a minimal polynomial among polynomials defined by a set of elements whose values coincide with each other when raised to the m-th power, and whose traces to a finite field of  $2^S$  elements are a value other than zero and coincide with each other in a subset of a finite field of  $2^{2^S}$  (where S represents a positive integer) elements, and the quotient polynomial when a polynomial defined by a subset of another finite field is divided by said minimal polynomial is a wiring connection polynomial that specifies the wiring connection in said polynomial multiplying units.

4. The error correction encoding apparatus as defined in claim 2, wherein wiring polynomials that specify the wiring connection in said polynomial dividing unit are randomly

selected, and each of (m-1) wiring polynomials that specify the wiring connection in said (m-1) polynomial multiplying units are determined by powers of said polynomials selected randomly that are different from each other.

5. An error correction encoding method using a low-density parity-check code, said method comprising;

in (m-1) polynomial multiplying units (where m represents an integer equal to or greater than two) respectively receiving a block having a length n (where n represents an integer equal to or greater than two) of an information bit string divided into said (m-1) blocks of bit strings having the length n and a single block of a bit string having a length (n-r) (where r represents an integer from 1 to n, inclusive), performing polynomial multiplication, and respectively outputting a bit series having the length n;

in an adder unit adding each output of said (m-1) polynomial multiplying units; and

in a polynomial dividing unit performing polynomial division of an output result of said adder unit and said block having the length (n-r) and outputs a redundant bit series having a length r.

6. The error correction encoding method as defined in claim 5, wherein a wiring polynomial that specifies the wiring connection in said polynomial dividing unit is a minimal polynomial among polynomials defined by a set of elements whose values coincide with each other when raised to the m-th power, and whose traces to a finite field of  $2^S$  elements are a value other than zero and coincide with each other in a subset of a finite field of  $2^{2^S}$  (where S represents a positive integer) elements, and the quotient polynomial when a polynomial defined by a subset of another finite field is divided by said minimal polynomial is a wiring polynomial that specifies the wiring connection in said polynomial multiplying units.

7. The error correction encoding method as defined in claim 5, wherein wiring polynomials that specify the wiring connection in said polynomial dividing unit are randomly selected, and each of (m-1) wiring polynomials that specify the wiring connection in said (m-1) polynomial multiplying units are determined by powers of said polynomials selected randomly that are different from each other.

\* \* \* \* \*